INFO370 Problem Set 1: Python, data manipulations (100 pt)

June 22, 2021

Instructions

This is the first problem set. These 100 points will give you 10 points of the final grade.

The goal of this problem set is to get you going with the basic python, such as creating and computing variables, and doing tricks with lists and dicts and functions. The good background reading is the Lubanovic (2014) book, chapters 2 (data types), 3 (lists, dicts, sets), 4 (code structures). The basics is also explained in python notes http://faculty.washington.edu/otoomet/machinelearning-py/python.html.

General requirements:

- All materials and resources that you use (with the exception of lecture slides) must be appropriately referenced within your assignment. In particular, note that Stack Overflow is licensed as Creative Commons (CC-BY-SA). This means you have to attribute any code you pick from SO (a link to the question/answer webpage will normally do).
- As the final submission, you should submit a) code; b) output; and c) explanations. If you are working with jupyter notebooks, all this can be easily included in the same notebook file but you still have to submit both your original file (so you grader can actually run the code), and an html version of it (which is much faster to check).
- Working together is fun and useful but you have to submit your own work. Discussing the solutions and problems with your classmates is all right but do not copy-paste their solution! First understand it, and thereafter create your own solution. Please list all your collaborators!

  1. ...
  2. ...

  ...
- If you cannot finish the question, you can still get partial for your incomplete attempts, given those are clean and readable enough. Attempt each question and to document your reasoning process even if you cannot quite get there!

Normally we ask you to write textual answers and explain the results. However, in this basic programming task this may not be necessary.

1 Computing (10pt)

Compute the following numbers, assign those to suitably named variables, and print:

1. How many seconds are there in year?

2. How many seconds is a typical human lifetime? Use the previously computed seconds in year.
3. What is your age in seconds? Compute this based on your age.
4. Create a logical variable that is true if you are more than 700M seconds old. Use logical operators, not if/else!

2 Lists (20pt)

1. Create a list 'movies' that contains the names of at least six movies you like
2. Using indexing and **slicing**, Create a list of three first movies in the list
3. Use slicing and list concatenation to create a list of the first two and the last two movies

3 Loops (10pt)

1. Create a list 'numbers' that is the numbers 70 through 79: in the following manner: create an empty list and add numbers to it in a loop (do not use list comprehension or 'list' function here!)
2. Use a loop to compute the mean value of the list: add the values to their sum in a loop, and at the end divide the sum by the number of the values
3. Use loop to compute sum of all numbers 1..100
4. Compute product of all numbers 1..100 (factorial, denoted by 100!)
5. Assign people to seats. Consider names *Adam*, *Ashin*, *Inukai*, *Tanaka*, and *Ikki*; and seats 33, 12, 45, 2 and 17. Print seat assignments by assigning each name to the corresponding seat. You coude should output

   Adam: 33
   Ashin: 12
   ...

   Hint: loop over the integer range of the length of names, and use indexing to access the corresponding name and seat number.

4 Dicts and sets (20pt)

1. Create a word frequency table using dicts that take a looong string (feel free to copy-paste some text here), splits it into individual words, and counts the number of occurrencies of each word (and prints the result). Let's ignore punctuation. Your code should run over individual words and increase the counter for that word, stored in a list. It has to check if a word exists in the dictionary, and if not, take an appropriate action.

   Hint: use the split() method to get words: `"I have no clue".split()` -> `['I', 'have', 'no', 'clue']`

   Hint 2: convert everything to lower case

Hint 3: use triple quotes """ to create long multi-line strings

Hint 4: proceed as you did with lists: first create an empty dict of word counts, thereafter loop over words, and increase the count for each word. But you have to handle the words that are not yet in the dict somehow.

2. Use *set* to find how many different words you have in your text above.

   Note: you can get this information from your word frequency dict too, but this task is about using set.

5 Comprehensions (10pt)

1. Use *list comprehension* to create a list of squares of numbers 1..10 (i.e. 1, 4, 9, ..., 100)
2. Use *dict comprehension* to create a dict of numbers 1..10 and their squares (i.3. 1:1, 2:4, ..., 10:100).

6 Functions (20pt)

1. Write a function that takes in time in the numeric form of HHMM (hours-minutes), and returns it in the numeric form of HH.HH (hours + fractions of hours). For instance, 1015 $\rightarrow$ 10.25 (10 hrs 15 mins $\rightarrow$ 10.25 hrs). Demonstrate it works using values 1015 and 345.

   The function should *return* (not print) the result as a *number*.

   Hint: use modulo operator % and integer division operator //. Modulo of 100 gives you minutes and integer division by 100 gives you hours

How much time did you spend?

And finally-finally, tell us how much time (how many hours) did you spend on this PS!

References

Lubanovic, B. (2014) *Introducing Python: Modern Computing in Simple Packages*, O'Reilly Media.